Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

ЛАБОРАТОРНА РОБОТА № 2

З дисципліни «Системне програмування»

На тему «Знайомство із середовищем розробки програм Microsoft Visual Studio»

ВИКОНАВ:
студент 2 курсу ФІОТ
групи ІО-82
Шендріков Є.О.

ПЕРЕВІРИВ:
ст. вик. Порєв В. М.

Київ – 2020

Мета:

Отримати перші навички роботи з Microsoft Visual Studio для створення програм, написаних мовою асемблера, а також вивчити команди MOV та CPUID.

Завдання:

1. Створити у середовищі MS Visual Studio проект з ім'ям **Lab2**. Встановити
необхідні параметри проекту – опції середовища розробки програм.

2. Написати вихідний текст програми на асемблері, додати файл вихідного
тексту у проект. Зміст вихідного тексту згідно з варіантом завдання.

3. Скомпілювати вихідний текст і отримати виконуємий файл програми.

4. Перевірити роботу програми. Налагодити програму.

5. Отримати дизасембльований текст машинного коду і проаналізувати його.

Усім студентам необхідно запрограмувати:
- початкове діалогове вікно-вітання від автора програми;
- виконання команди CPUID з параметрами 0, 1, 2 а також 80000000h, 80000001h, 80000002h, 80000003h, 80000004h, 80000005h та 80000008h. Кожний результат виконання CPUID команди потрібно виводити у окремому діалоговому вікні. Якщо результати CPUID утворюють текстові дані, то виводити їх як рядки тексту. Отримати дизасембльований код і проаналізувати його. Пояснити значення N-го біту кожного результату команди CPUID, де N – номер студента у списку у журналі. Для пояснення використати документ «Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 2A: Instruction Set Reference», доступний на сайті фірми Intel.

Текст програми:

```
.586
.model flat, stdcall

include \masm32\include\kernel32.inc
include \masm32\include\user32.inc

includelib \lib\kernel32.lib
includelib \lib\user32.lib

.data
	CaptionFirst db "Lab 2", 0
	TextFirst db "Jack Shendrikov, IO-82", 0
	Text db 'EAX=xxxxxxxx', 13, 10,
	     'EBX=xxxxxxxx', 13, 10,
	     'ECX=xxxxxxxx', 13, 10,
	     'EDX=xxxxxxxx', 0

	res dd 256 dup(0)

	Caption0 db "CPUID 0",0
	Caption1 db "CPUID 1",0
```

```
	Caption2 db "CPUID 2",0

	Caption00 db "CPUID ..00h", 0
	Caption01 db "CPUID ..01h", 0
	Caption02 db "CPUID ..02h", 0
	Caption03 db "CPUID ..03h", 0
	Caption04 db "CPUID ..04h", 0
	Caption05 db "CPUID ..05h", 0
	Caption08 db "CPUID ..08h", 0

	Model db 32 dup(0)
	CaptionModel db "CPUID 0 - Processor name", 0

.code

DwordToStrHex proc
	push ebp
	mov ebp,esp
	mov ebx,[ebp+8]
	mov edx,[ebp+12]
	xor eax,eax
	mov edi,7
@next:
	mov al,dl
	and al,0Fh
	add ax,48
	cmp ax,58
	jl @store
	add ax,7
@store:
	mov [ebx+edi],al
	shr edx,4
	dec edi
	cmp edi,0
	jge @next
	pop ebp
	ret 8
DwordToStrHex endp

main:
	invoke MessageBoxA, 0, ADDR TextFirst, ADDR CaptionFirst, 0

	mov eax, 0
	cpuid
	mov dword ptr[Model], ebx
	mov dword ptr[Model+4], edx
	mov dword ptr[Model+8], ecx
	invoke MessageBoxA, 0, ADDR Model, ADDR CaptionModel, 1h

	mov eax, 0
	cpuid
	mov dword ptr[res], eax
	mov dword ptr[res+4], ebx
	mov dword ptr[res+8], ecx
	mov dword ptr[res+12], edx
	push [res]
	push offset [Text+4]
	call DwordToStrHex
	push [res+4]
	push offset [Text+18]
	call DwordToStrHex
	push [res+8]
	push offset [Text+32]
	call DwordToStrHex
	push [res+12]
	push offset [Text+46]
	call DwordToStrHex
	invoke MessageBoxA, 0, ADDR Text, ADDR Caption0, 1h
```

```
mov eax, 1
cpuid
mov dword ptr[res], eax
mov dword ptr[res+4], ebx
mov dword ptr[res+8], ecx
mov dword ptr[res+12], edx
push [res]
push offset [Text+4]
call DwordToStrHex
push [res+4]
push offset [Text+18]
call DwordToStrHex
push [res+8]
push offset [Text+32]
call DwordToStrHex
push [res+12]
push offset [Text+46]
call DwordToStrHex
invoke MessageBoxA, 0, ADDR Text, ADDR Caption1, 1h

mov eax, 2
cpuid
mov dword ptr[res], eax
mov dword ptr[res+4], ebx
mov dword ptr[res+8], ecx
mov dword ptr[res+12], edx
push [res]
push offset [Text+4]
call DwordToStrHex
push [res+4]
push offset [Text+18]
call DwordToStrHex
push [res+8]
push offset [Text+32]
call DwordToStrHex
push [res+12]
push offset [Text+46]
call DwordToStrHex
invoke MessageBoxA, 0, ADDR Text, ADDR Caption2, 1h

mov eax, 80000000h
cpuid
mov dword ptr[res], eax
mov dword ptr[res+4], ebx
mov dword ptr[res+8], ecx
mov dword ptr[res+12], edx
push [res]
push offset [Text+4]
call DwordToStrHex
push [res+4]
push offset [Text+18]
call DwordToStrHex
push [res+8]
push offset [Text+32]
call DwordToStrHex
push [res+12]
push offset [Text+46]
call DwordToStrHex
invoke MessageBoxA, 0, ADDR Text, ADDR Caption00, 1h

mov eax, 80000001h
cpuid
mov dword ptr[res], eax
mov dword ptr[res+4], ebx
mov dword ptr[res+8], ecx
mov dword ptr[res+12], edx
push [res]
```

```
push offset [Text+4]
call DwordToStrHex
push [res+4]
push offset [Text+18]
call DwordToStrHex
push [res+8]
push offset [Text+32]
call DwordToStrHex
push [res+12]
push offset [Text+46]
call DwordToStrHex
invoke MessageBoxA, 0, ADDR Text, ADDR Caption01, 1h

mov eax, 80000002h
cpuid
mov dword ptr[res], eax
mov dword ptr[res+4], ebx
mov dword ptr[res+8], ecx
mov dword ptr[res+12], edx
push [res]
push offset [Text+4]
call DwordToStrHex
push [res+4]
push offset [Text+18]
call DwordToStrHex
push [res+8]
push offset [Text+32]
call DwordToStrHex
push [res+12]
push offset [Text+46]
call DwordToStrHex
invoke MessageBoxA, 0, ADDR Text, ADDR Caption02, 1h

mov eax, 80000003h
cpuid
mov dword ptr[res], eax
mov dword ptr[res+4], ebx
mov dword ptr[res+8], ecx
mov dword ptr[res+12], edx
push [res]
push offset [Text+4]
call DwordToStrHex
push [res+4]
push offset [Text+18]
call DwordToStrHex
push [res+8]
push offset [Text+32]
call DwordToStrHex
push [res+12]
push offset [Text+46]
call DwordToStrHex
invoke MessageBoxA, 0, ADDR Text, ADDR Caption03, 1h

mov eax, 80000004h
cpuid
mov dword ptr[res], eax
mov dword ptr[res+4], ebx
mov dword ptr[res+8], ecx
mov dword ptr[res+12], edx
push [res]
push offset [Text+4]
call DwordToStrHex
push [res+4]
push offset [Text+18]
call DwordToStrHex
push [res+8]
push offset [Text+32]
call DwordToStrHex
```

```
        push [res+12]
        push offset [Text+46]
        call DwordToStrHex
        invoke MessageBoxA, 0, ADDR Text, ADDR Caption04, 1h

        mov eax, 80000005h
        cpuid
        mov dword ptr[res], eax
        mov dword ptr[res+4], ebx
        mov dword ptr[res+8], ecx
        mov dword ptr[res+12], edx
        push [res]
        push offset [Text+4]
        call DwordToStrHex
        push [res+4]
        push offset [Text+18]
        call DwordToStrHex
        push [res+8]
        push offset [Text+32]
        call DwordToStrHex
        push [res+12]
        push offset [Text+46]
        call DwordToStrHex
        invoke MessageBoxA, 0, ADDR Text, ADDR Caption05, 1h

        mov eax, 80000008h
        cpuid
        mov dword ptr[res], eax
        mov dword ptr[res+4], ebx
        mov dword ptr[res+8], ecx
        mov dword ptr[res+12], edx
        push [res]
        push offset [Text+4]
        call DwordToStrHex
        push [res+4]
        push offset [Text+18]
        call DwordToStrHex
        push [res+8]
        push offset [Text+32]
        call DwordToStrHex
        push [res+12]
        push offset [Text+46]
        call DwordToStrHex
        invoke MessageBoxA, 0, ADDR Text, ADDR Caption08, 1h


        invoke ExitProcess, 0
end main
```

Аналіз результатів

Інсталяція програми Microsoft Visual Studio прошла успішно, завдання з другої лабораторної роботи виконано без помилок. Програма є елементарним прикладом на мові програмування Assembler.

Висновок

У ході виконання лабораторної роботи було закріплено на практиці навички використання Microsoft Visual Studio, а також було написано просту програму, яка відобразила інформацію про процесор комп'ютера. Я отримав перші навички роботи з для створення програм, написаних мовою асемблера, а також вивчив команди MOV та CPUID.